머신러닝 & 딥러닝 7

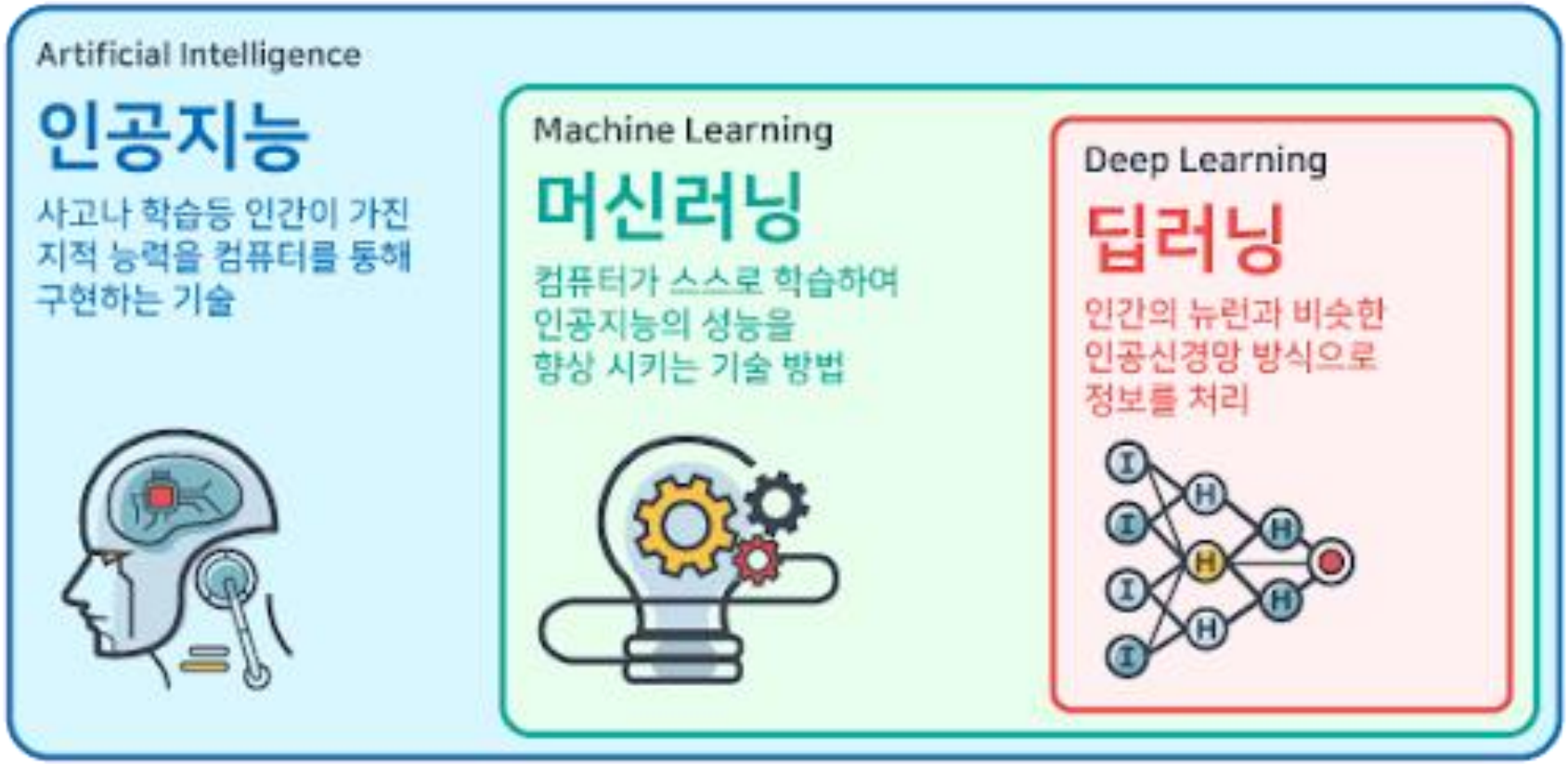
AI 학술동아리 <MLP>

- Index

- 이미지 데이터 정규화 하기

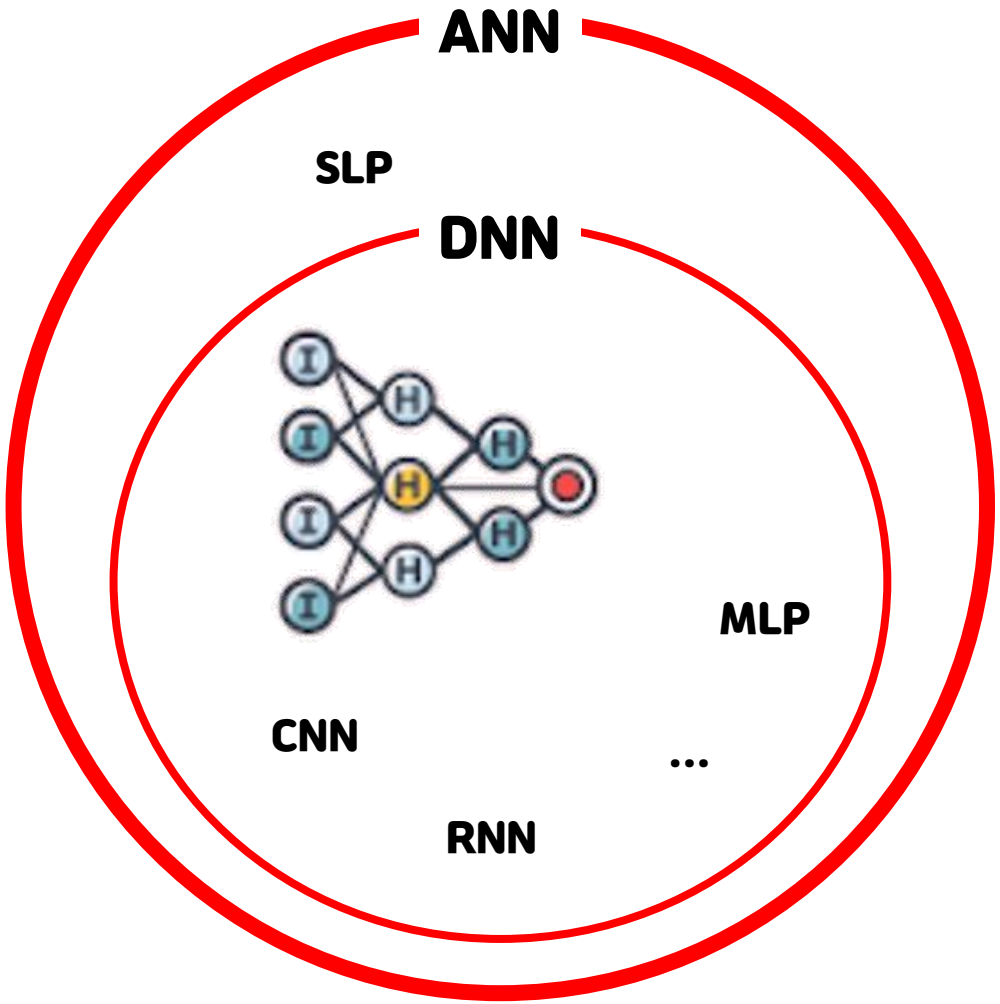
이미지의 픽셀은 0~255 사이의 정숫값을 가짐
-> 이 경우 보통 255로 나누어 0~1 사이의 값으로 정규화
(표준화는 아니지만 양수 값으로 이루어진 이미지를 전처리할 때 널리 사용하는 방법)

- Artificial Intelligence(인공지능)



DL 라이브러리는 다른 ML 라이브러리와 다르게 GPU사용해서 ANN훈련
- GPU는 벡터와 행렬 연산에 매우 최적화

1. Artificial Neural Network(인공 신경망)



ANN을 줄여서 NN(Neural Network)라고도 함

ANN을 DL이라고도 함

ANN(Artificial Neural Network) = 인공 신경망

DNN(Deep Neural Network) = 심층 신경망
= Deep Learning

-> Layer가 여러개

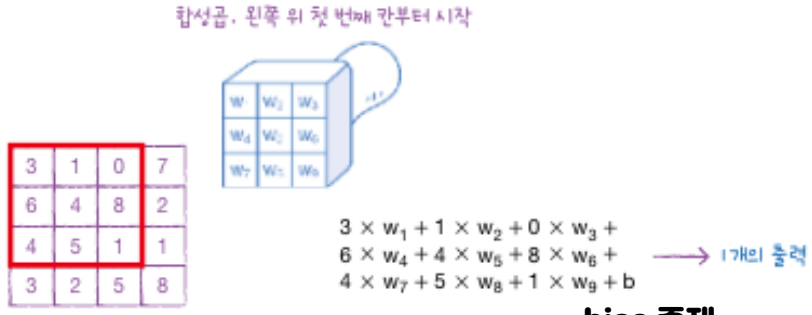
SLP(Single Layer Perceptron) = 단층 퍼셉트론

MLP(Multi Layer Perceptron) = 다층 퍼셉트론

보통 kernel size는 (3, 3), (5, 5)를 이용!

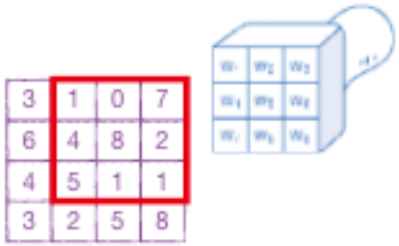
1. Convolution Neural Network(합성곱 신경망)

convolution 1

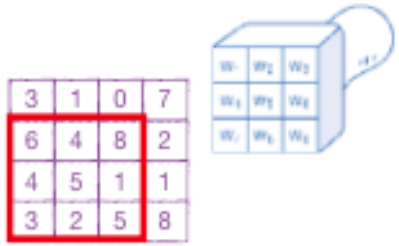


bias 존재

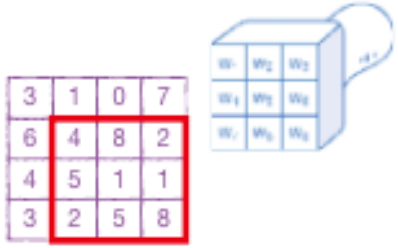
convolution 2



convolution 3



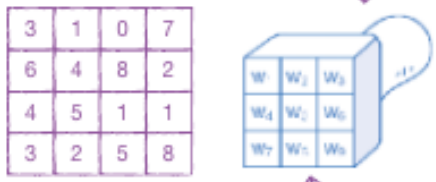
convolution 4

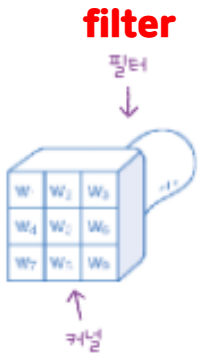


filter ⊃ kernel

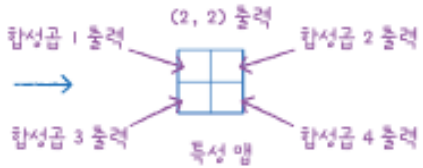
도장 weight, bias

보통은
filter = kernel
로 쓰임





kernel



feature map

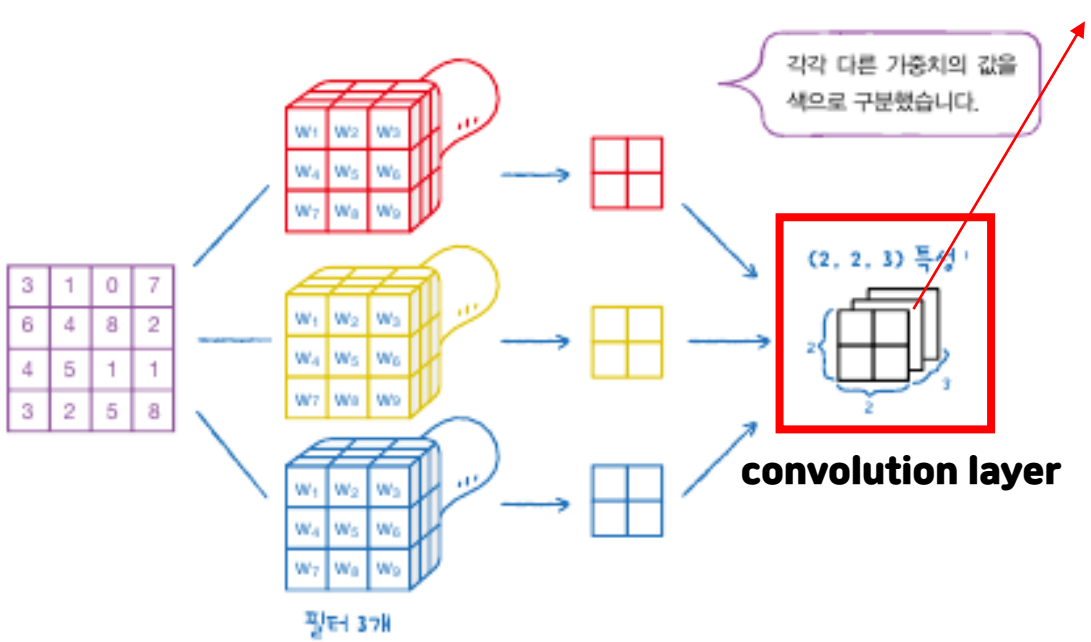
(4, 4) -> (2, 2)

kernel 크기는 hyperparameter

filter는 왼쪽에서 오른쪽으로, 위에서 아래로 이동

dense layer와 동일하게 단순히 input과 weight를 곱하지만, 2차원 형태를 유지하는 점이 다름

1. CNN



feature map들

convolution layer

filter의 개수도 hyperparameter

일반적으로 1개 이상의 convolution layer를 가진 NN을 CNN이라고 부름

CNN은 특히 이미지에 있는 특징을 찾아 압축하는 데 뛰어난 성능을 냄

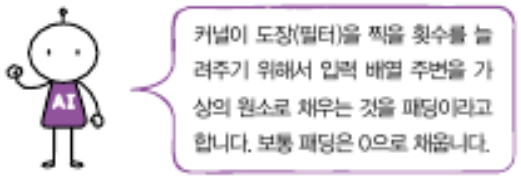
기본 convolution 과정을 거쳐 feature map을 만드는 경우를 valid padding이라고 함

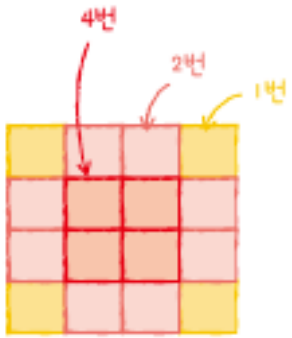
보통 same padding을 많이 이용!

- Padding(패딩)

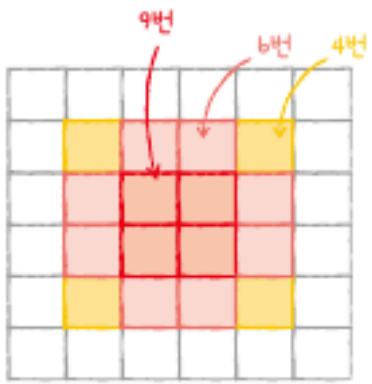
convolution 과정을 거치고도 output의 크기를 input과 동일하게 만들 때 이용
input 배열 주위를 가상의 원소로(0으로) 채움 = same padding
0으로 채우기 때문에 zero padding

0	0	0	0	0	0
0	3	1	0	7	0
0	6	4	8	2	0
0	4	5	1	1	0
0	3	2	5	8	0
0	0	0	0	0	0



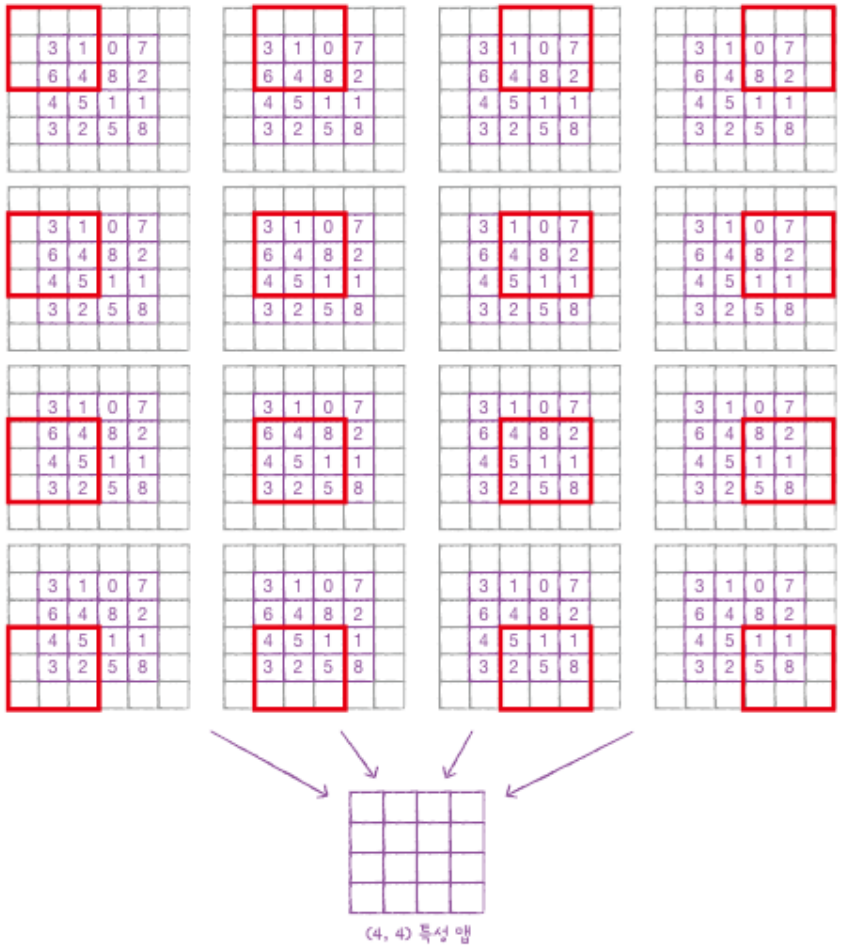


(4:2:1)



(9:6:4)

패딩 여부에 따라 각 픽셀이 convolution에 참여하는 비율이 크게 차이 남
-> 비율 차이를 줄이기 위해 same padding 이용



보통 stride size는 1을 많이 이용!

- Stride(스트라이드)

convolution 연산을 기본적으로 좌우, 위아래로 한 칸씩 이동
이런 이동의 크기를 stride(스트라이드)라고 함

-> 두 칸씩(N 칸씩) 이동 가능
=> (5, 5) 크기의 입력에 (3, 3) 크기의 커널을 적용하면 결과는 (2, 2) 크기의 출력(그림으로 그려보기)

1보다 큰 stride를 사용하는 경우는 드묾

보통 max pooling을 많이 이용!

average pooling은 feature map에 있는
중요한 정보를 (평균하여) 희석시킬 수 있기 때문

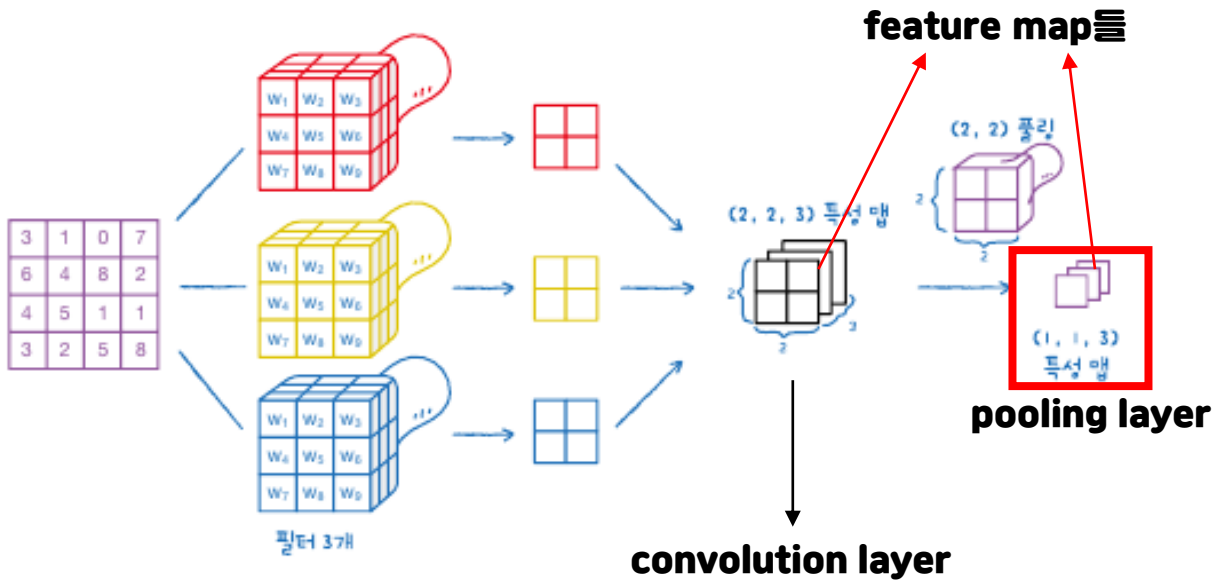
보통 (2, 2) pooling을 사용해 절반으로 줄임!

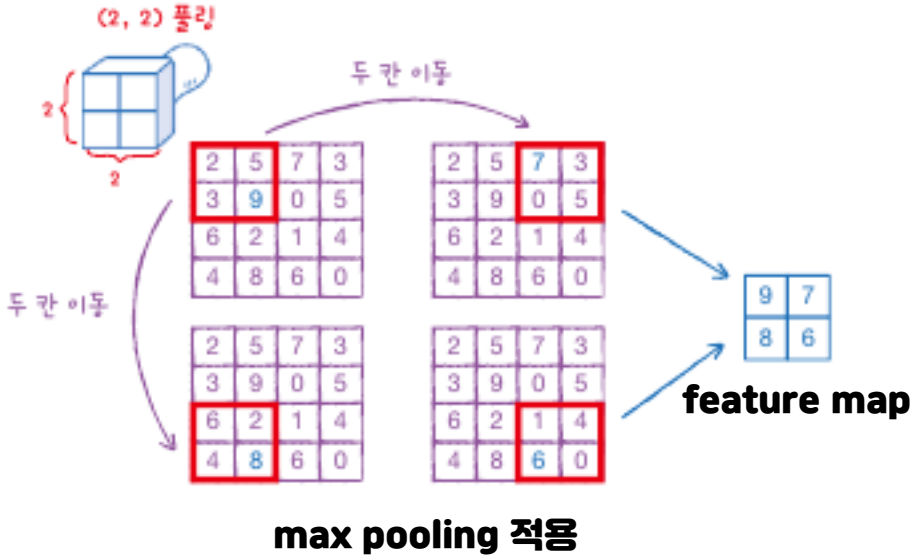
- Pooling(풀링)

convolution layer에서 만든 feature map의 가로세로 크기를 줄이는 역할
- feature map의 개수(filter 개수만큼 생성됨)는 줄이지 않음(예: (2, 2, 3) -> (1, 1, 3))

pooling에는 가중치가 없음, 도장을 찍은 영역에서 가장 큰 값을 고르면 max pooling(최대 풀링),
평균값을 계산하면 average pooling(평균 풀링)

convolution과 다르게 kernel이 겹치지 않고 이동(예: pooling의 크기가 (2, 2)이면 stride는 2,
pooling의 크기가 (3, 3)이면 stride는 3)

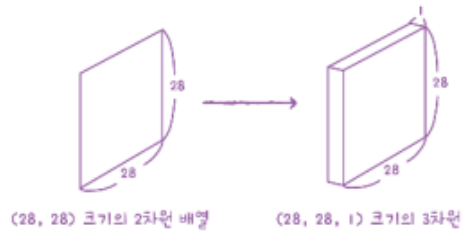




보통 convolution에서 stride를 크게 하여 feature map의 크기를 줄이는 것보다
pooling을 통해 feature map의 크기를 줄이는 것이 더 나은 성능을 보임

1. CNN(흑백 이미지)

보통 convolution layer와 pooling layer은 거의 항상 함께 사용

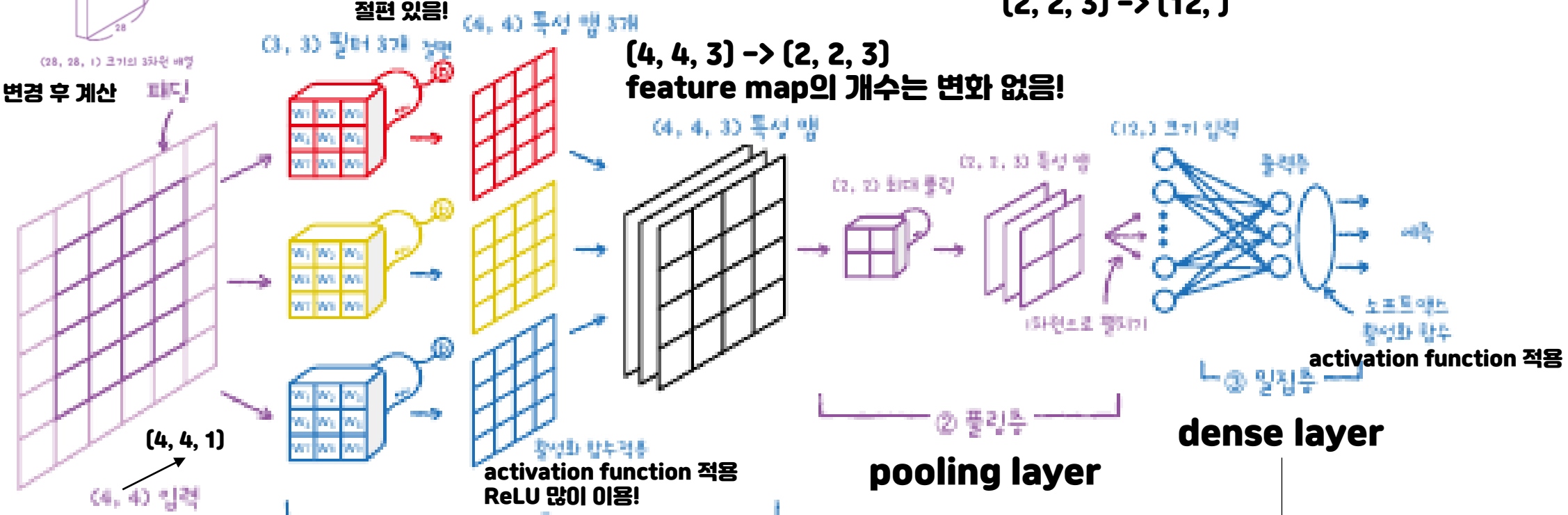


사실 차원을 1로 변경 후 계산

절편 있음!

(2, 2, 3) -> (12,)

(4, 4, 3) -> (2, 2, 3)
feature map의 개수는 변화 없음!



activation function 적용
ReLU 많이 이용!

activation function 적용

input layer

convolution layer

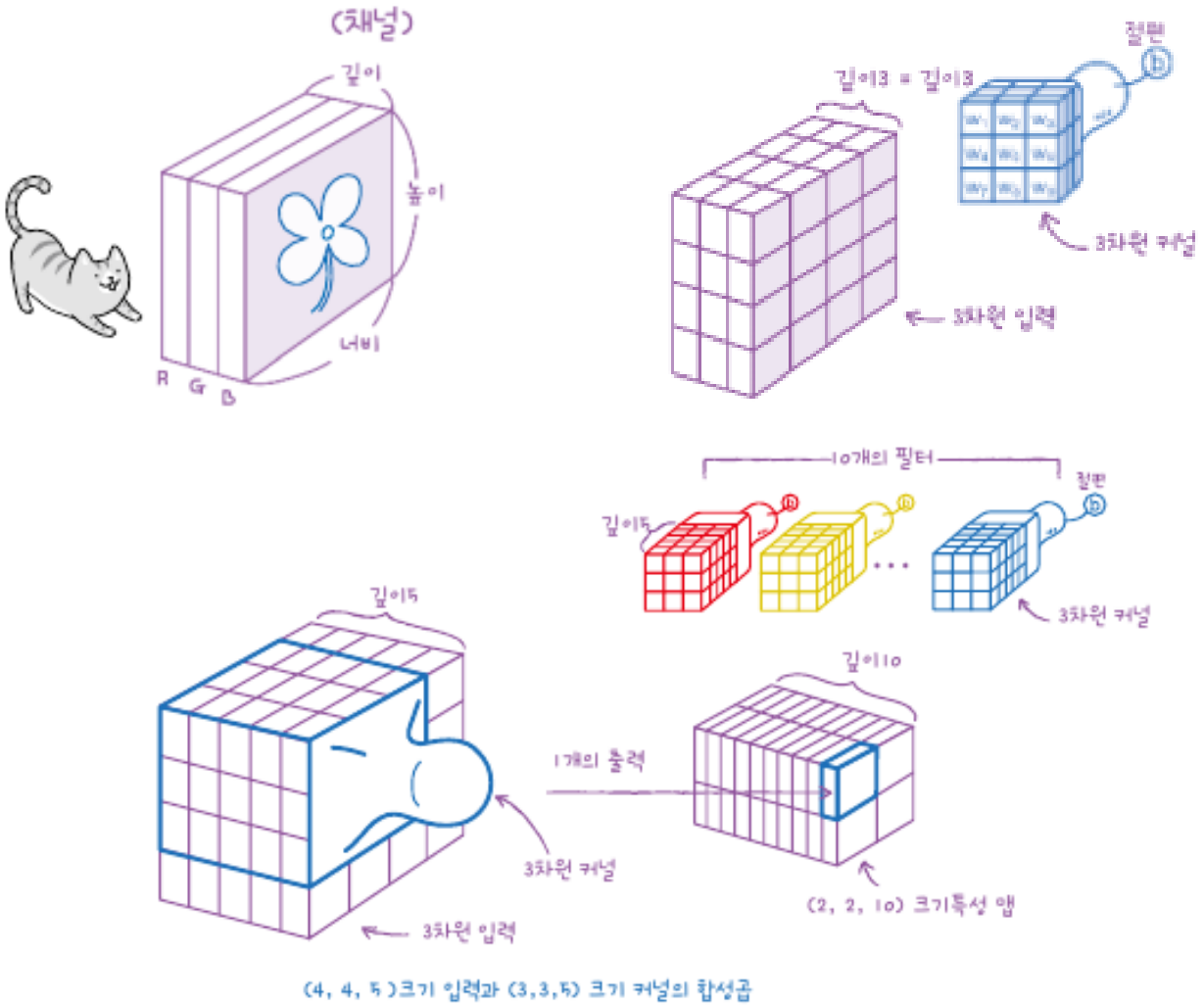
pooling layer

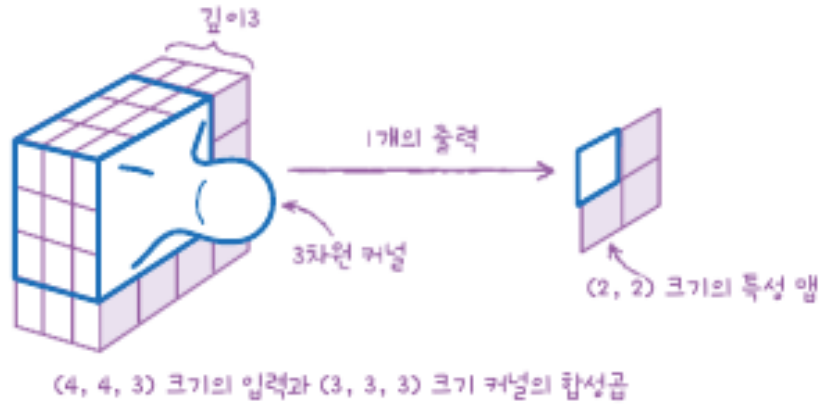
dense layer

이 부분을 그냥 FC Layer라고도 함
또는 MLP라고 부르기도 함

input 이미지는 항상 깊이(채널) 차원이 있어야 함

1. CNN(컬러 이미지)





kernel의 깊이는 input의 깊이와 같음

CNN은 너비와 높이는 점점 줄어들고
깊이는 점점 깊어지는 것이 특징

- CNN 문제 1

- **문제 1:**

  어떤 convolution layer가 컬러 이미지에 대해 5개의 filter를 사용해 same padding으로 convolution을 수행합니다. 그 다음 (2, 2) polling layer를 통과한 feature map의 크기가 (4, 4, 5)입니다. 이 경우 convolution layer에 주입된 input의 크기는 얼마인가요?

- **정답 1:**

  변화 과정 : -> ->

- CNN 문제 2

- **문제 2:**

  다음과 같은 input에서 (3, 3) kernel과 valid padding으로 convolution을 수행합니다.
  filter의 개수는 1개이고 input의 깊이(채널)도 1개입니다. bias는 0이라고 가정합니다.
  이 convolution의 결과를 계산해 보세요.

(5, 5) 입력

3	0	9	1	2
5	1	2	0	7
8	2	4	1	3
2	1	5	3	6
4	1	6	2	7

(3, 3) 커널

2	0	1
2	0	1
2	0	1

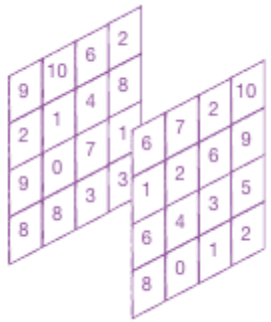
- **정답 2:**

  - 계산 과정 생략

- CNN 문제 3

- **문제 3:**

  다음과 같은 (4, 4, 2) 크기의 feature map이 있습니다.
  (2, 2) max polling의 결과를 계산해 보세요.



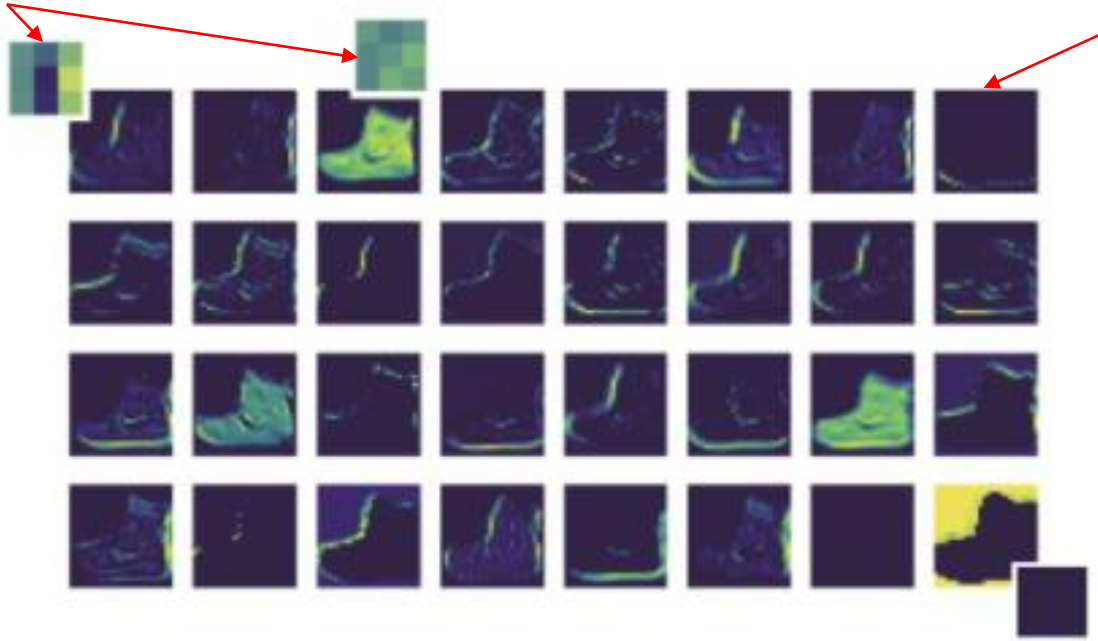
- **정답 3:**

  - 계산 과정 생략

- Visualization(시각화)

weight visualization(가중치 시각화)

feature map visualization(특성 맵 시각화)



이를 **시각화**하면 CNN 동작 원리에 대한 통찰을 키울 수 있음